REPORT DOCUMENTATION PAGE	Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE Technical Report	3. DATES COVERED (From - To) -

4. TITLE AND SUBTITLE Steganography Detection Using Entropy Measures	5a. CONTRACT NUMBER W911NF-11-1-0174
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER 206022
6. AUTHORS Dr. Alfredo Cruz (Advisor), Eduardo Melendez	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAMES AND ADDRESSES Polytechnic University of Puerto Rico 377 Ponce De Leon Hato Rey San Juan, PR 00918 -	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) U.S. Army Research Office P.O. Box 12211 Research Triangle Park, NC 27709-2211	10. SPONSOR/MONITOR'S ACRONYM(S) ARO
	11. SPONSOR/MONITOR'S REPORT NUMBER(S) 58924-CS-REP.26

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT
There are two problems in steganalysis: (1) detecting the existence of a hidden message and (2) decoding the message. As terrorist groups have been known to use steganography in planning their attacks, this has become an important problem of national security.
This research proposal is only concerned with the first problem of hidden message detection using steganalysis [8]. The approach is to statistically analyze the least significant bit(s) of each color dimension of

15. SUBJECT TERMS
Steganalysis, entropy, jpeg files

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON Alfredo Cruz
a. REPORT UU	b. ABSTRACT UU	c. THIS PAGE UU	UU		19b. TELEPHONE NUMBER 787-622-8000

Report Title

Steganography Detection Using Entropy Measures

ABSTRACT

There are two problems in steganalysis: (1) detecting the existence of a hidden message and (2) decoding the message. As terrorist groups have been known to use steganography in planning their attacks, this has become an important problem of national security. This research proposal is only concerned with the first problem of hidden message detection using steganalysis [8]. The approach is to statistically analyze the least significant bit(s) of each color dimension of each pixel to look for some kind of a pattern. In the absence of a hidden message this should look like random noise. Addition of a hidden message will affect the entropy of the data [4]. This difference should be detectable by comparing the entropy of unaltered picture files with the entropy of files with embedded steganography. For this research freely available software for embedding hidden messages will be used to create sample image files to analyze. For the statistical analysis R Language will be used. The following steps will be followed in the work plan:
□Obtain sample jpegs from the Internet or other source □ Import these sample files as data files into R Language □ Statistically analyze least significant bits. □ Use steganography to hide messages in a sample of jpeg files. □ Import as a data file into R language and statistically analyze the least significant bits of the jpeg files with known hidden messages.□ Compare with original file in terms of entropy.

Steganography Detection Using Entropy Measures

Report 1

By Eduardo Meléndez

Universidad Politécnica de Puerto Rico

August 19, 2012

DoD HBCU # W911NF-11-1-0174:

"Enhancing Research in Networking & System Security, and Forensics, in Puerto Rico"

Alfredo Crúz, PhD
Jeff Duffany, PhD
Eduardo Meléndez, MS

meléndez.eduardo@gmail.com

Contents

1 Introduction

There are two problems in steganalysis: (1) detecting the existence of a hidden message and (2) decoding the message. As terrorist groups have been known to use steganography in planning their attacks, this has become an important problem of national security. This research proposal is only concerned with the first problem of hidden message detection using steganalysis [8].

The approach is to statistically analyze the least significant bit(s) of each color dimension of each pixel to look for some kind of a pattern. In the absence of a hidden message this should look like random noise. Addition of a hidden message will affect the entropy of the data [4]. This difference should be detectable by comparing the entropy of unaltered picture files with the entropy of files with embedded steganography. For this research freely available software for embedding hidden messages will be used to create sample image files to analyze. For the statistical analysis R Language will be used.

The following steps will be followed in the work plan:

- Obtain sample jpegs from the Internet or other source
- Import these sample files as data files into R Language
- Statistically analyze least significant bits.
- Use steganography to hide messages in a sample of jpeg files.
- Import as a data file into R language and statistically analyze the least significant bits of the jpeg files with known hidden messages.
- Compare with original file in terms of entropy.

2 Least Significant Bit

One of the advantages of Steganography alone, is the fact that the message does not attract attention to itself. No party should have knowledge of the existence of a message, but the sender and the recipient. Once the message has been compromised, steganography simply fails [9].

2.1 Image definition

Historically, many strategies has been used to hide messages sent between two or more parties. Currently, one of the means to hide a message are images. The mechanism is to embed a message digitally in a picture by manipulating the bits representing the different colors [9]. Let us assume that a picture is depicted in the different scales of gray, including black and white (monochrome and grayscale images). Each pixel is represented by a string of 8 bits. From black to white, we have $2^8 = 256$ different tones of gray. These range from the representation of white, 00000000, through black, 11111111. A given message with proper size can be embedded in a cover (image) by manipulating the bits on each pixel. Let us assume that in a particular pixel, the gray is represented by 00001111. By switching the second bit we obtain a new binary string, 01001111. The latter change has modified the original picture.

For the case of 24 bits images, digital colour pictures (RGC colour model pictures), each color red, green and blue is represented by a string of 8 bits, so that each pixel can be colored by manipulating each color (each string of bits) [9]. In each pixel there can be 256 representations for each color, so that there are $256^3 = 16,777,216$ possibilities of color shades. Below, three sets of 8 bits-string are defined. Each set represents a color. From an original image, say

$$00001010 \quad 00110101 \quad 00011110$$

we can change the 4^{th} bit from left to right, obtaining

$$00011010 \quad 00100101 \quad 00001110$$

2.2 Image Compression

When working with large images, we start having problems handling large files. Some sort of compression is necessary in order to better handle these images. There are two types of compression: lossy and lossless [9]. An example of the first type of compression technique is JPEG (Joint Photographic Experts Group) image format. For the second type, we have the GIF (Graphical Interchange Format) and the 8-bit BMP (Microsoft Windows Bitmap file). In the first case loss of information occurs, while in the second the integrity of the original information remains intact. The technique of steganography implemented will depend on the compression technique used [9].

2.3 Least Significant Bit

The object of steganography is to prevent suspicion upon the existence of a message, regardless of the mean used. Small changes in the tone of gray will be imperceptible to the human eye. The Least Significant Bit (LSB) is a simple approach to modify an image, while at the same time, making the change imperceptible to the human eye. By considering the redundant bits (least significant bits), imperceptible changes take place by changing the 8^{th} bit in the string of eight bits. For example, by changing 00001111 to 00001110, we have applied the least significant bit technique.

2.4 Significant Bit Image Depiction

Steganography fails to comply in its purpose, at the very moment when the existence of the message has been compromised. Even when steganography is not infallible, its strength lies entirely on the non-knowledgeable of its containment, whatever it is. When the mean of communication is a picture, from which a text or a message can be extracted, its infallibility is directly related to the manipulation of the pixels. In particular, by manipulating the LSB, any message is safe as long as it remains imperceptible to the human eye. The following pictures show the level of *visual perception* in relation to the change of bits of each channel (color) in each pixel. An image is compared before and after the LSB technique has been applied. A simple procedure (switching all LSB for each channel in each pixel) has produced a different image that simply cannot be distinguished from the original one.

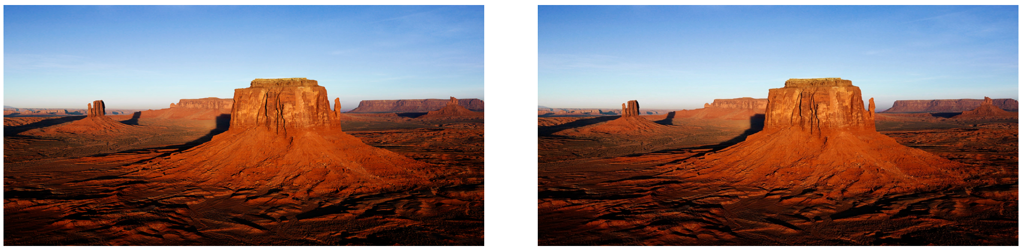
Figure 1: Original Depiction (left); LSB Depiction (right)

We are unable to perceive any changes in the image after the LSB technique.

Below, the resultant images after switching bits number 2 and 3, start to show gradual changes in color shades.

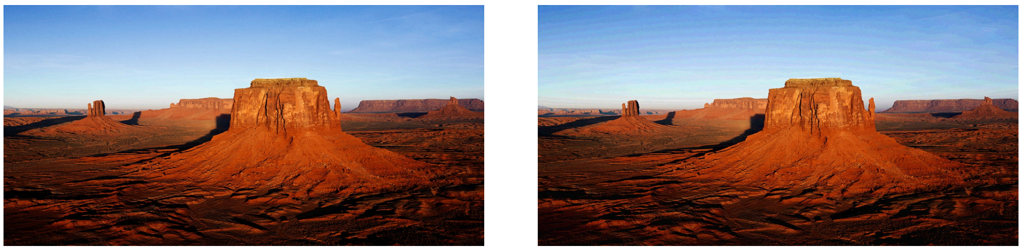
Figure 2: Switch of bit number 2 (left); Switch of bit number 3 (right)

Changes made from the bits number 4 and on are definitely evident. Colors are distorted and degraded.

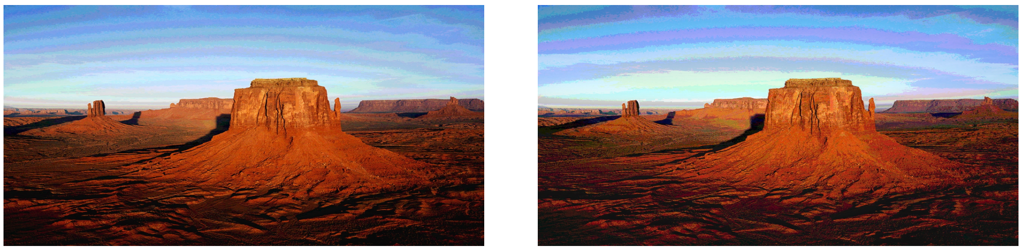
Figure 3: Switch of bit number 4 (left); Switch of bit number 5 (right)

Finally, the switch of bit number 8 in all channels for every pixel makes the modification evident. So it is that it can be perceived the by human eye. These bits (number 8) are extremely significant, and if steganography is the intended purpose, would be unwise to choose bit number 8. Below, we show the original image side by side with the 8^{th}-bit-switch depiction.

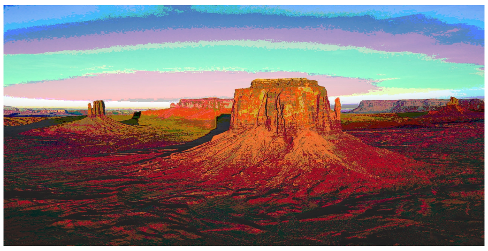
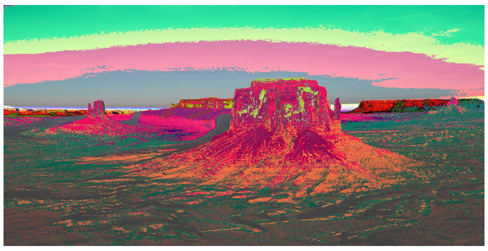

Figure 4: Switch of bit number 6 (left); Switch of bit number 7 (right)

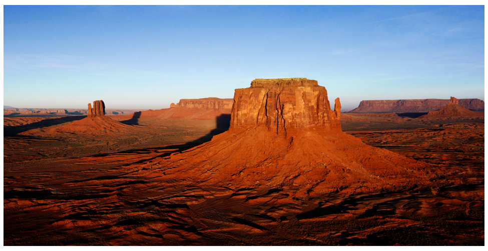
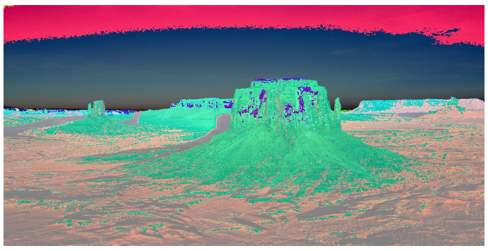

Figure 5: Original depiction (left); Switch of bit number 8 (right)

2.5 R-Codes

In the previous section we compared an original image with different modifications, by having a bit switched over. We have managed to accomplish this using the R-language (2.14,2.15). We have loaded packages jpeg, ReadImage, boolfun, RGraphics and many others. The two crucial packages were ReadImages and boolfun to use read.jpeg and toBin, respectively. The first allows us to read an image in jpg or jpeg format. The second one gives an integer binary representation. Another function used from boolfun is toInt, which returns an integer from a given binary representation.

The function below, imageManipulation2, switches the i^{th} bit for each channel in every pixel of an image in jpeg or jpg format. When the picture is read, a 3-dimensional array is built. Each array contains the numeric representation for each color in every pixel. The values in this matrix are real numbers between 0 and 1. These numbers are normalized and they are of the form $n/255$, where $n = 0, 1, ..., 255$. These number are multiplied by 255 resulting in an integer ranging from 0 to 255, precisely 256 values. The integers are converted to their binary representation. A particular bit is switched over for each binary. This sequence is converted to an integer and normalized. The resulting matrix is an image different from the original.

```
%x is the image
%This function switches the bit at position n
%N is the length of the binary representation
%t is a title
%normalization (division by 255)

function (x,n,N,t)
{
    nrow <- dim(x)[1]
    ncol <- dim(x)[2]
    ncha<- dim(x)[3]
    for(i in 1:ncha)
    {
        for(j in 1:nrow)
        {
            for(k in 1:ncol)
            {
                if(n<=N && n>=1)
                {
                        m = x[j,k,i]*255
                        %toBin returns a binary representation of length N
                        y = toBin(m,N)
                        y[n] = !y[n]
                        %toInt returns an integer from a binary representation of a certain length
                        x[j,k,i] = toInt(y)/255
                }
            }
        }
    }
    plot(x,main=t)
    x
}
```

3 Entropy

Great part of the development of the Mathematical Theory of Communications during the sixties is due to Claude Shannon, a Bell Labs Mathematician. Shannon is one of the founders of the Information Age. Shannon made clear that uncertainty is the marketable item produced to satisfy wants or needs in a society of communication.

The amount of information or uncertainty, output by an information source is a measure of its entropy. The entropy of an information source *S*, according to Shannon is defined as

$$H(S) = \sum_i p_i log(1/p_i), \tag{1}$$

where p_i is the probability that S_i in S will take place. The factor $log(1/p_i)$ indicates the amount of information contained in S_i, i.e., the number of bits needed to code S_i.

For example, in an image with uniform distribution of gray-level intensity, i.e., $p_i = 1/256$, then the number of bits needed to code each gray level is 8 bits. The entropy of this image is 8.

4 Steganalysis

There are many approaches to detect steganographic images by using statistical tools. In this section we are going to mention some of them briefly. There are many tests that can be done upon certain statistical properties. These tests go from very simple to more complex and sophisticated [8]. Westfeld and Pfitzmann observed that embedding encrypted data into a GIF image changes the histogram of its color frequencies [8]. One property of binary data, in the context of encryption, is that zeros and ones are equally likely. The LSB technique has the property that in an image with an embedded encrypted data, if one color A, happens more often than color B, color A is changed more often than color B, rather than the other way around. This results in a reduction of color frequency between colors A and B, because of the embedding. A simple diagram (histogram) can depict these differences, and visually identified.

Another approach taken in measuring statistical properties is by analyzing the frequency of the DCT (Discrete Cosine Transformation) coefficients. By comparing the empirical distribution and a theoretic distribution, a χ^2 test can be used. The latter is a traditional view in using statistical theories. However, we must mention briefly that by analogy, the same statistical tools used in the $frequentist$ approach have their counterparts in the Bayesian statistics [1].

Now, if we consider, without further pretensions the bits-changes in a sequence mode, other statistical tools might be useful. Such an interpretation might appeal to Time Series [3] and from a Bayesian point of view, to Dinamical System.

5 Steganographic Systems and Detection Frameworks

Many steganographic systems can be used for embedding. From these systems, we can mention JSteg, JSteg-Shell, JPHide and OutGuess. Most of them work around the concept of DCT coefficients. As a detector counterpart, we can mention Stegdetect.

6 R-Language

Several statistical tools for image processing are available of all sort. Packages have been developed to process, manipulate and analyze images in many languages, such as C, C++, Matlab, R, etc.. Particularly, the latter provides an environment where many packages for image processing and binary manipulation can be used. From these packages we can mention DICOM, ANALYZE, NIFTI, ReadImages, RGraphics, jpeg, bmp, png, boolfun, caTools, bindata, bit, boolean, biOps, biOpsGUI and pixmap.

7 Conclusion

There are several steps to take from here. First, in order to better understand and comprehend the theories discussed (briefly), we must reproduce some of the results provided by the articles. Second, the use of statistical software tools are rather known. We have identified the R-language for implementation. Third, we must select a picture for processing, bits manipulation, color and DCT coefficients frequency and distribution. Fourth, use one of the software packages for steganography or have one developed in a simply and rudimentary manner (as an option). Fifth, determine the entropy of the picture (before and after embedding). Sixth, have the picture submitted to statistical analysis, steganalysis. Seventh, develop or identify a mechanism to analyze in mass, pictures from the Internet or other source provided.

The project is in an on going status.

8 Reference

1. Aruna Ambalavanan and Rajarathnam Chandramouli, “A Bayesian Image Steganalysis Approach to Estimate the Embedded Secret Message”

2. Christopher Jaynes, Susan Landau and Allen Hanson, “Communication and Secrecy: Issues in Digital Steganography”, COINS Technical Report 96-77, October, 1996

3. David Lowe, “Steganography, Time Series and Interesting Components”, NCRG, Aston University, UK

4. Hafiz Malik, K.P. Subbalakshmi and R. Chandramouli, “Nonparametric Steganalysis of QIM Data Hiding using Approximate Entropy”

5. James C. Judge, “Steganography: Past, Present, Future”, Sans Institute InfoSec Reading Room

6. Muthiyalu Jothir, Navaneetha Krishnan, “Statistical Models for Secure Steganography Systems”, May 15, 2006

7. Niels Provos, “Defending Against Statistical Steganalysis”, Center for Information Technology Integration, University of Michigan, Pp. 323-336 of the Proceedings of the 10^{th} USENIX Security, Symposium, August 13-17, Washington

8. Niels Provos and Peter Honeyman, “Detecting Steganographic Content on The Internet”

9. T. Morkel, J.H.P. Eloff and M.S. Oliver, “An Overview of image Steganography” Information and Computer Security Architecture (ICSA) Research Group, Department of Computer Science, University of Pretoria, 0002, Pretoria, South Africa

10. “Color Models”, http://www.aphic.com/datainflux/colour/ColorModels_adobe.pdf

11. "Package bindata" from R-Language, February 14, 2012

12. "Package biOps" from R-Language, February 14, 2012

13. "Package bit" from R-Language, February 14, 2012

14. "Package boolean" from R-Language, February 14, 2012

15. "Package boolfun" from R-Language, February 14, 2012

16. "Package caTools" from R-Language, May 23, 2012

17. "Package jpeg" from R-Language, Frebruary 14, 2012

18. "Package pixmap" from R-Language, February 15, 2012

19. "Package ReadImages" from R-Language, April 19, 2012

20. "Package RGraphics" from R-Language, August 17, 2012